Projet ARGOSI

Dossier préliminaire du second incrément
Version 1.0

Ident_Doc **:** LIN:T1.6:DC:A:1.0

Préparé par :
Pascal André
LINA - 2, rue de la Houssinière
BP 92208
44322 NANTES cedex 3 (France)

Tel : +33.02.51.12.59.65
Fax : +33.02.51.12.58.12
email : Pascal.Andre@univ-nantes.fr
http://www.sciences.univ-nantes.fr/info/perso/permanents/andre

Résumé :

Ce document est le dossier préliminaire de conception du second incrément du projet ARGOSI. Il résume le premier incrément (analyse et critique) réalisé par un groupe d'étudiants en projet et présente la planification du second incrément, c'est-à-dire les objectifs, tâches, rôles, estimations et répartitions des travaux ainsi qu'un planning initial de notre travail.

Date de création : 16/10/2004

Date de dernière révision : 25/11/2004

Documents référencés :

LIN:T1.1:DC:A:1.1
LIN:T1.5:NT:A:1.0
LIN:T1.1:DC:A:0.1
LIN:T2.5:SP:A:1.0

I. Introduction

Dans le cadre du développement du second incrément pour le projet ARGOSI (document LIN:T1.1:DC:A:1.1), il est prévu la remise d'un dossier préliminaire du second incrément, dont le plan nous a été remis en séance de TP.

Ce dossier fait la liaison entre les deux incréments, c'est-à-dire qu'il synthétise le travail réalisé dans le premier incrément et prépare le travail du second incrément. La synthèse se traduit par une analyse (section II) des produits mis à notre disposition (dossier d'analyse LIN:T2.5:SP:A:1.0, rapport de projet des étudiants d'IUP Miage 3 IUP:T7:NT:B:3.0, code des étudiants). Cette analyse de l'existant nous conduit à une critique de l'existant (section III), qui met en évidence les points forts et les points faibles du développement du premier incrément. Enfin, la section III décrit la préparation du second incrément, qui se conçoit en quatre parties : objectifs principaux, découpage en tâches, rôles des participants, planification (répartition des tâches sur les rôles et dans le temps).

Ce document est clairement un contrat pour la gestion de ce projet.

II. Analyse de l'existant

Le projet ARGOSI dans son ensemble consiste en la mise en place d'un système d'information automatisé relatif à la gestion de l'association (administration) et de ses activités (séminaires et pôles). L'existant de ce projet est le résultat (les produits) du premier incrément, à savoir un premier document (LIN:T2.5:SP:A:1.0) contenant l'analyse des besoins, l'analyse et la conception préliminaire, un second document (IUP:T7:NT:B:3.0) contenant des éléments relatifs à la conception et la réalisation, et enfin un code Java.

II.1 Analyse des besoins et analyse

L'analyse du cas ARGOSI (document LIN:T2.5:SP:A:1.0) comprend une analyse des besoins, une analyse et quelques éléments de la conception. L'ensemble est documenté à l'aide de la notation UML. L'analyse des besoins met en évidence les besoins fonctionnels de l'application, clairement séparés en trois parties : administration, séminaires et pôles. Cette analyse des besoins est volontairement incomplète : le premier incrément de l'analyse ne vise qu'une des trois parties principales, la partie Administration, et plus particulièrement la gestion des membres puisque la gestion des situations est ignorée (section II, p. 15, LIN:T2.5:SP:A:1.0). Dans la Figure 1, nous avons mis en évidence, par partie analysée, vis-à-vis de l'ensemble de l'application. Les intersections montrent que les parties ne sont pas complètement indépendantes.

L'analyse des besoins affine la spécification fonctionnelle de la gestion des membres par différents cas d'utilisation, illustrés par des scénarios. Ici aussi la description est incomplète : tous les cas d'utilisation et tous les scénarios ne sont pas explorés. Un modèle du domaine complète la description des cas d'utilisations, qui permet de fixer les informations principales manipulées. La description est précisée par les contraintes OCL.

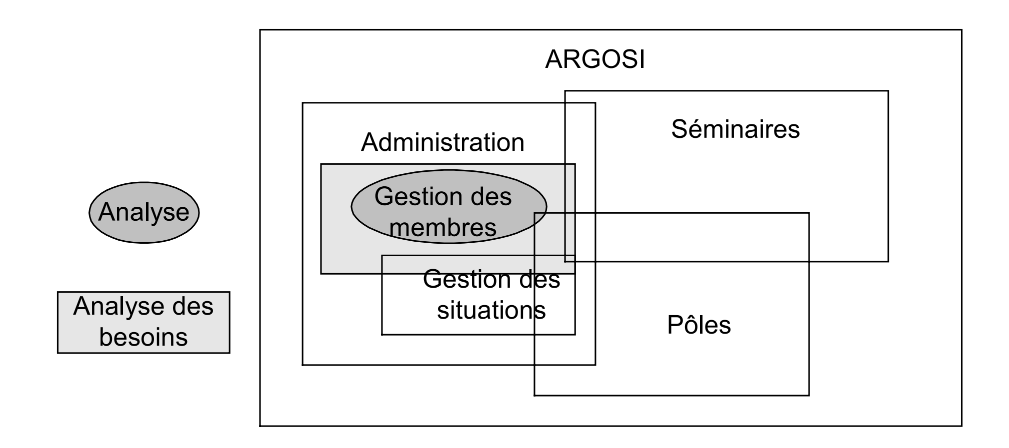


Figure 1- Cadrage de lanalyse des besoins et de l'analyse du cas Argosi

L'analyse reprend une partie des cas d'utilisation et des scénarios sous forme de séquences et de collaborations qui mettent en évidence la structure logique du système en faisant apparaître de nouveaux objets, liés à l'interface ou au contrôle de l'application, et en affinant les objets métiers du modèle du domaine. Compte-tenu du type d'application, il a été choisi une spécialisation des objets en trois catégories : interface, contrôle, métier (p. 20, LIN:T2.5:SP:A:1.0). Il en résulte un nouveau modèle des objets métiers, support de la base de données.

En résumé : une description volontairement incomplète mais qui fixe les idées.

II.2 Conception

La conception préliminaire est abordée superficiellement dans le document LIN:T2.5:SP:A:1.0. Elle indique simplement que l'application est structurée en une base de donnée et une ou plusieurs applications, relatives à la partie Administration.

Les principes de conception sont étudiés plus finement dans le rapport de projet des étudiants d'IUP Miage 3 (IUP:T7:NT:B:3.0). Puisqu'il ne s'agit pas d'un rapport technique, il ne se focalise pas uniquement sur la conception et la réalisation mais inclut des éléments de gestion de projet, de documentation et d'appréciation personnelle. Nous devons donc commencer par extraire l'information pertinente. Nous la structurons dans les sous-sections suivantes.

II.2.1 Architecture du logiciel

Le logiciel comprend principalement une application Java et une base de données relationnelle pour les objets persistants. Il s'agit donc d'une architecture client/serveur simplifiée.

II.2.2 Contraintes techniques

Le cahier des charges implique l'usage du langage de programmation Java pour développer l'application. Cela permet de rester dans le monde objet entre l'analyse et la réalisation. La conception préliminaire prévoyant un stockage des objets persistants dans une base de données, il a été choisi le SGBD Oracle pour des considérations pratiques liées à l'usage du système d'exploitation Linux (p. 7, IUP:T7:NT:B:3.0).

II.2.3 Outils de développement

Outre le SGBD Oracle, les étudiants ont opté pour l'EDI (Environnement de développement intégré) Eclipse (p.7, IUP:T7:NT:B:3.0).

II.2.4 Limites de la conception

La conception reprend la partie de l'analyse relative à la gestion des membres. La conception prévoit (p.7, IUP:T7:NT:B:3.0) :

1) La gestion des **adhérents**. « Il s'agit ici d'enregistrer de nouveaux adhérents, de modifier leurs caractéristiques comme leur adresse par exemple, de radier des adhérents, de définir le bureau ou de démettre un membre de ce bureau, d'éditer des listes de membres. »
2) La gestion des **assemblées générales**. « Il s'agit de gérer les convocations des membres à l'assemblée générale et gérer les élections du bureau. Il nous faut programmer une assemblée Générale (AG), convoquer une AG et organiser le système de vote.
3) La gestion des réunions. « La gestion des réunions consiste en la programmation de celles-ci, en la convocation des membres à celles-ci, en la composition du bureau, en la demande de démission de membres du bureau et en l'édition de la liste des membres du bureau. »

II.2.5 Principes de conception

Les principes de conception sont relatifs à l'implantation de toutes les classes en Java et des classes métier dans une base de données. Il y a donc une représentation double pour les classes métier. Le chargement peut se faire à la demande ou globalement. La seconde option a été choisie car elle est plus simple à mettre en œuvre (p.12, IUP:T7:NT:B:3.0). La communication entre l'application et la BD se font via JDBC.

II.2.6 Détails de la conception

La conception (appelée analyse conceptuelle) reprend la partie de l'analyse relative à la gestion des membres. La conception prévoit (p.11, IUP:T7:NT:B:3.0) :

- Application Java : « Nous implémentons les classes principales Membre, Réunion, FonctionCA et les classes d'ensembles respectives Ens_FonctionCA, Ens_Membre et Ens_Reunion. En revanche, pour la classe démission, nous ne jugeons pas utile de constituer une classe mais nous ajoutons un champ démission pour l'attribut état de type énuméré EnumM. Nous ajoutons des classes permettant la gestion de l'interface comme les classes Frm_oper, Frm_table et Frm_Main.»
- Base de données : « Nous reprenons également le diagramme des classes pour constituer la base de données Oracle. Les classes se transforment principalement en des tables (membre, fonctionCA, réunion). Nous ressentons alors le besoin d'en ajouter d'autres. Pour les types énumérés EnumC, EnumD, EnumF, EnumM, EnumR, de nouvelles tables sont ainsi constituées avec une clé primaire se rapportant au champ énuméré. Une autre table convocation est aussi créée de façon à réaliser la convocation d'un membre à une réunion. »

En fait peu de détails sont fournis.

Une description succincte des interfaces est donnée sous forme de copie d'écrans et de commentaires (p.12, IUP:T7:NT:B:3.0). L'interface est rudimentaire : le menu principal est donné dans une fenêtre individuelle et comprend la gestion des membres, des fonctions et des réunions. Ces trois parties ont des interfaces similaires : affichage, ajout, modification, suppression, effacement et recherche plus ou moins évoluée, soit des fonctions de navigation dans des listes (des ensembles ici).

II.2.7 Apports méthodologiques

Les étudiants ont travaillé sur deux points :

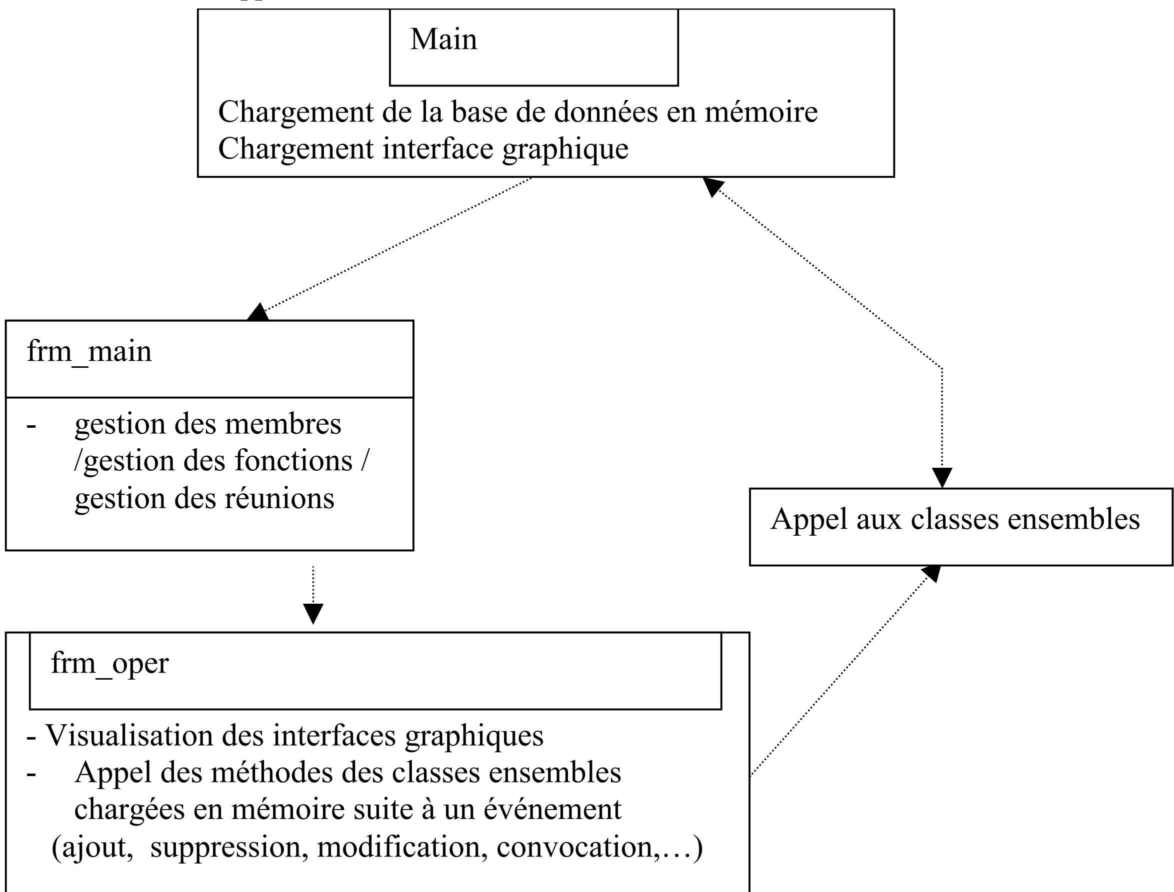
- La traduction de UML à Java (annexe 1.a, IUP:T7:NT:B:3.0).
- La traduction de UML à un SGBD relationnel (annexe 1.b, IUP:T7:NT:B:3.0).

II.3 Réalisation et test

La réalisation (appelée implémentation) reprend la partie de la conception (en fait de l'analyse) relative à la gestion des membres. La réalisation prévoit (p.15, IUP:T7:NT:B:3.0) :

- La Gestion des Membres. « Dans cette partie de l'application, nous pouvons retrouver un membre de l'association à partir d'une recherche sur le nom de l'entreprise. Plusieurs membres peuvent travailler dans la même entreprise. L'accès à ceux-ci se fera par les boutons suivant et précédent. L'ajout, la modification et la suppression de membres sont possibles. Pour certain d'entre eux, des fonctions leur sont attribuées. »
- La Gestion des Fonctions du Bureau. « Au cours de sa vie dans l'association, un membre peut occuper plusieurs fonctions dans le bureau : président, secrétaire, trésorier… La gestion du bureau incorpore des possibilités d'accès aux membres assurant une fonction à une date donnée et permet également de retrouver la réunion au cours de laquelle la fonction a été attribuée. L'ajout et la modification de fonctions sont possibles. »
- La Gestion des Réunions. « La recherche d'une réunion se fait par la date. On pourra observer toutes les réunions qui ont lieu le même jour, les membres qui sont convoqués et ajouter des convocations si nécessaire. L'ajout et la modification de réunions sont possibles. »

L'articulation de l'application est la suivante :



Il n'est pas fait mention de test organisé.

II.4 Test et contrôle du code de l'application

Un manuel d'utilisation est fourni (annexe 2, IUP:T7:NT:B:3.0). Pour fonctionner (compilation réussie), il faut que la connexion à la base de donnée soit établie. Autrement dit, il faut la créer manuellement sous Oracle.

Un manuel de programmation est fourni (annexe 3, IUP:T7:NT:B:3.0), qui inventorie différentes fonctions non-implantées : « Les dates sont actuellement sous forme de String et aucun contrôle sur leur cohérence n'est réalisé. La gestion des AG et des membres candidats n'est pas réalisée. La gestion des votes n'est pas réalisée. La gestion des demandes de démission n'est pas réalisée. L'interface

graphique actuelle permet de réaliser des tests mais ne correspond pas à(aux) interface(s) graphique(s) nécessaire(s) aux utilisateurs. »

Le test fonctionnel doit en principe s'appuyer sur la description des cas d'utilisation et les scénarios (test boîte noire), on peut l'affiner en contrôlant les diagrammes de séquence.

Le contrôle du code se fait par des lectures.

II.5 Gestion de projet

Les aspects gestion de projet recouvrent des éléments relatifs à :

- L'organisation et la planification : les étudiants se sont organisés en deux groupes, selon l'architecture logicielle, base de donnée et application Java.
- Les normes : les étudiants suivent la norme de documentation du projet (LIN:T1.5:NT:A:1.0) et une codification du nommage des variables (p. 8, IUP:T7:NT:B:3.0).
- La formation : au moins un étudiant n'était pas formé à UML.
- Le suivi : des réunions hebdomadaires ont permis de recentrer le projet et prendre les décisions nécessaires.

Le départ du chef de projet a pénalisé le déroulement du projet (p. 18, IUP:T7:NT:B:3.0).

Les objectifs du premier incrément n'ont pas été atteints.

III. CRITIQUES

Dans cette partie, l'objectif est de mettre en évidence des lacunes, des erreurs, ou simplement des points faibles du premier incrément. Plusieurs approches sont possibles : on reprend la structure de l'analyse de l'existant point par point, ou bien on recadre en fonction des objectifs généraux du second incrément, ou alors on établit mixage des deux approches ou encore on définit une structure ad-hoc. On choisit cette dernière approche.

On ne traite pas ici de la gestion de projet, considérant qu'il s'agit d'un autre développement, avec une autre équipe.

III.1 Couverture des besoins fonctionnels

Le passage de l'analyse des besoins à la réalisation d'un prototype a été une succession de réductions sévères de l'espace de travail. En effet, seuls les gestions des membres, des fonctions et des réunions sont abordées, qui plus est partiellement. Les fonctions rudimentaires d'ajout, modification, suppression et recherche ont été implantées. Les listes ne sont pas implantées. Les liens entre ces différentes « gestions » sont implantés. En résumé, le premier incrément n'atteint pas, à notre avis une couverture suffisante pour bien évaluer l'architecture et l'environnement technique. Bien sûr, s'agissant d'un premier incrément, il est indiqué clairement dans l'analyse la limitation (volontaire) des besoins fonctionnels. Cependant, la conception et la réalisation qui la reflète (un peu trop : est-ce une rétro-conception ?) marquent plus une restriction due à un manque de temps (une conséquence due aux difficultés de mise en œuvre) qu'un choix de conception. Cela n'est pas mis en évidence dans le rapport.

III.2 Validation de l'architecture et des choix techniques

Les contraintes techniques prévoient l'usage de la BD, non pas comme une partie de l'application, mais un support à la persistance. De ce fait, l'application n'est pas simplement une interface d'accès à la BD mais un programme autonome, qui doit pouvoir fonctionner sans BD, avec un système de sauvegarde différent. Compte-tenu de cette remarque, deux choix de conception sont à discuter :

- Liaison application et BD. Dans la conception, il a été choisi un chargement complet plutôt qu'un chargement à la demande, car plus facile à implanter dans un premier temps. Nous pensons qu'il serait plus intéressant de mettre en œuvre une politique de chargement à la demande, sous forme de transactions BD.
- Choix du SGBD. Le SGBD Oracle a été choisi parce que les étudiants le connaissaient et en disposait pour le projet, parce qu'il permet de développer l'application sur Linux ou Windows en Java. Il n'est pas crucial de bénéficier de traitements pointus sur la BD, hormis la persistance, c'est pourquoi la plupart des SGBD conviennent, qu'ils soient relationnels ou pas. En l'absence de contraintes précises à ce sujet, il nous semble important de bénéficier d'une offre BD multi-plateforme (Unix, Windows), ce qui élimine l'offre propriétaire Access de Microsoft. On privilégiera un SGBD gratuit pour l'instant.

Un point important est la cohérence entre les classes métiers de l'application et leur représentation dans la BD (classes/tables, types/domaines, contraintes…). Pour l'instant, les deux sont implantés séparément (cf. les méthodes de traduction de UML à Java (annexe 1.a, IUP:T7:NT:B:3.0) et de UML aux relations (annexe 1.b, IUP:T7:NT:B:3.0). La vérification de cohérence est manuelle lors des séances de mise en correspondance entre les deux parties de l'application. On pourra affiner les méthodes d'implantation mais surtout établir un protocole (automatisé si possible) pour vérifier la conformité. On note par exemple, des problèmes de contraintes (*DELETE CASCADE*) dans les tables.

Pour l'instant, une architecture 2/3 a été conçue et implantée. Il n'a pas été discuté d'une architecture en 3 tiers, type web, envisageable pour une association dont les membres sont distants. Il s'agit là d'un objectif de l'application, à discuter avec le « client » Argosi.

Les étudiants ne donnent pas de références pour les outils utilisés et les versions dans leur rapport. Le choix d'éclipse peut se révéler intéressant puisqu'un module UML peut y être ajouté. Il nous semble important de mettre en évidence des outils pour la génération de documentation.

III.3 Analyse et conception

L'analyse présente assez clairement la situation et nous semble une bonne introduction au projet ARGOSI. Un des intérêts se situe dans les commentaires sur la méthode de travail. Son principal défaut est la complétude. Tous les cas d'utilisation des parties sélectionnées ne sont pas décrits de même que tous les scénarios des UC choisies. Il faut donc compléter cette description fonctionnelle et la poursuivre dans l'analyse et la conception.

La conception est quasi inexistante et semble se résumer à la réalisation. Il nous semble important de mettre en évidence cette étape clé, qui met en correspondance l'architecture technique, liée aux choix techniques, et l'architecture logique issue de l'analyse et surtout de mettre clairement en évidence les choix de conception, leur argumentation pour qu'on puisse ensuite les contrôler et les valider. Cette remarque vaut tant pour le SGBD et son interface que pour l'IHM.

D'un point de vue fonctionnel, il manque la gestion des situations (depuis l'analyse), l'historique des fonctions, les élections, etc.

III.4 Programmation de l'application

Le choix du langage Java est imposé, liberté est donnée pour l'environnement de développement (voir choix techniques) et les bibliothèques de classes. La réalisation est à l'image de la conception : inachevée. De nombreuses lacunes sont à déplorer, relatives à :

- la couverture fonctionnelle,
- les interfaces graphiques,
- la méthode et la documentation du codage.

Détaillons ces différents points.

III.4.1 Couverture fonctionnelle

La couverture fonctionnelle insuffisante et surtout pas assez mise en évidence : un tableau récapitulant ce qui est prévu, ce qui est fait serait apprécié.

III.4.2 Interfaces graphiques

Les interfaces sont rudimentaires, peu agréables et peu ergonomiques (on doit deviner à quoi les boutons correspondent et dans quel ordre on doit procéder). Par exemple les recherches ne sont pas intuitives, les répercussions des suppressions ne sont précisées, les attributs à fournir pour les requêtes sont à deviner… On préfère une interface avec une barre de menus et différents modes d'aide. On peut s'inspirer des applications courantes sous Linux ou Windows. Ce point nous semble à revoir.

III.4.3 Méthode et codage

Comme dans la plupart des projets, la mise en œuvre et surtout sa documentation, les dernières étapes du développement sont loin d'être finalisés. Les principaux reproches sont les suivants :

- Documentation insuffisante malgré un manuel utilisateur et un manuel de référence. Il faut tester soi-même te lire le code pour comprendre.
- Hormis une norme de nommage des variables (pas toujours appliquée), aucune règle de programmation n'est mise en évidence.
- Peu de commentaires dans le code et pas toujours pertinents.

III.5 Qualité, vérification, validation et tests

Comme pour toute application, il est important de penser à la qualité des spécifications et du logiciel produit [AV 01]. En particulier la validité, la robustesse, la facilité d'utilisation pour l'utilisation et les qualités suivantes pour les développeurs l'extensibilité, la réutilisabilité et la portabilité.

Concernant la facilité d'utilisation, nous avons déjà critiqué les IHM. Pour la validité et la robustesse, on fait appel à la vérification et au test.

La vérification n'a pas été abordée elle concerne l'étude des propriétés des produits (modèle d'analyse, de conception, programme) et du processus (cohérence entre les étapes : quel lien entre l'analyse et la conception, la conception et l'implantation, etc.). En particulier, les fenêtres des interfaces ne reflètent pas les scénarios de l'analyse des besoins ou de l'analyse.

Il n'est pas fait mention d'une validation des utilisateurs.

Le rapport ne fait pas mention d'un test organisé.

- Pas de plan de test
- Pas de précisions sur le traitement des erreurs (exception, messages…). Pas de règle adoptée.

Concernant les principales qualités pour le développement, la portabilité est induite par l'usage de java et d'un SGBD multi-plateformes. L'extensibilité, induite par le développement à objet, est à mesurer lors du développement du second incrément, elle est liée à la documentation du code et aux critères de groupement modulaires dans les classes de l'interface et du contrôle.

Nous n'avons aucune garantie sur l'usage du code, ni d'aide pour l'appréhender (variables inutilisées, champs de la BD inexploités).

Les modules ne semblent pas conçus en vue d'une réutilisation.

III.6 Documentation

La documentation est inégale. Notons que le rapport de projet IUP est un condensé de la documentation « normale » du projet : dossier de conception, notes techniques, compte-rendu des réunions… Tout cela ne nous était pas fourni.

La présentation des documents devrait être normalisée selon le modèle décrit dans le document LIN:T1.5:NT:A:0.1.

Le code est assez peu documenté et sa documentation n'est pas normalisée. Les modules ne semblent pas conçus en vu d'une réutilisation.

IV. MISE EN PLACE DU SECOND INCRÉMENT

IV.1 Objectifs

L'objectif principal du second incrément est de valider l'architecture logicielle. Cela sous-entend plusieurs aspects à affiner et valider :

- Articulation entre l'application Java et la base de données : mettre au point les mécanismes de communication, choisir au besoin un nouveau SGBD, un environnement de développement adapté…
- Mise au point des interfaces : présentation, bibliothèques de classes Java, ergonomie…
- Compléter la prise en compte des besoins et étudiant la bonne intégration des nouveaux éléments à l'application existante : réutilisabilité et incrémentalité.

D'autres objectifs sont affichés :

- Maintenir le code du premier incrément : corriger les erreurs, ajouter les fonctionnalités prévues dans l'analyse pour la gestion des membres.
- Analyser et éventuellement concevoir et réaliser la partie Gestion des séminaires.
- Tester
- Réutiliser

Quelques pistes d'étude pour valider l'architecture et les choix techniques

- Autre choix de BD, de système IHM
- Tests d'architecture 3 tiers

IV.2 Tâches

Décomposer les objectifs en tâches selon la nomenclature des activités indiquées dans le rapport LIN:T1.5:NT:A:1.0.

Estimer les charges et les incidences entre tâches.

Détailler les produits du développement (documentations, codes…)

IV.3 Rôles

Nommer les participants, leur cursus et leurs connaissances

Donner les rôles des participants au projet

Dépend de l'équipe de projet

IV.4 Planification

Planifier les tâches en fonction des rôles et du temps.

Planifier les risques et leur récupération

Jalons obligatoires

V. GLOSSAIRE ET ABBRÉVIATIONS

Cette partie est incomplète.

BD	Base de données
IHM	Interface homme/machine. Modèle d'interface graphique de l'application.
SGBD	Système de Gestion de Bases de Données
UC	Use Case, cas d'utilisation (UML)
UML	Unified Modeling Language

VI. BIBLIOGRAPHIE ET DOCUMENTATION

[AV 01] *Conception des systèmes d'information ; Panorama des méthodes et des techniques*, Pascal André et Alain Vailly, Editions ellipses, 2001, ISBN 2-7298-0479-X.

[AV 03] *Exercices corrigés en UML ; Passeport pour une maîtrise de la notation*, Pascal André et Alain Vailly, Editions ellipses, 2003, ISBN 2-7298-1725-5.

[RUP 99] *The Unified Software Development Process*, James Rumbaugh et al., Addison-Wesley, 1999, ISBN 0-201-57169-2.

[CLO 99] *RUP, XP, Architectures et outils* de Pierre-Yves Cloux, Dunod, 2003, ISBN 2-100-06430-4.

[RV 03] *UML en action*, de Pascal Roques et Franck Vallée, Eyrolles, 2e édition, 2003, ISBN 2-212-11213-0.

Documentation ARGOSI

LIN:T1.1:DC:A:1.1	Présentation du projet IUP-Miage 3
LIN:T1.5:NT:A:1.0	Normes de présentation des documents
LIN:T2.5:SP:A:1.0	Analyse du cas ARGOSI
LIN:T1.4:NT:A:1.0	Synthèse de cours UML
IUP:T7:NT:B:3.0	Rapport de projet IUP Miage 3 2003-2004
LIN:T1.6:DC:A:1.0	Dossier préliminaire du second incrément
LIN:T1.4:DD:A:0.2	Plan du dossier de conception